AMENDMENTS TO THE SPECIFICATION

Please replace paragraph [0002], with the following rewritten paragraph:

[0002] This application claims the benefit of U.S. Provisional Application ~~Applications~~ No. 60/395,871, filed 07/12/2002, which is herein incorporated in its entirety by reference.

Please replace paragraph [0028], with the following rewritten paragraph:

[0028] End-users have greater choices and can easily implement "pay as you go" features. There is device independence with scalable hardware that enhances compatibility ~~compatiblity~~ on a global scale. The manufacturers and also reap great benefits by reusing stock hardware and having lesser number of models. The network providers simply service offerings without having to support a large number of different protocols and standards.

Please replace paragraph [0058], with the following rewritten paragraph:

[0058] When inserted into a host, the module adds the capability of multiple radio types, determined by the module's programming. In some embodiments, this may be a software defined single function radio, dual function radio (DFR) or multi-function radio. In some embodiments, the module possesses sufficient hardware and software resources to simultaneously instantiate a plurality of different radio signal types (waveforms), such as IEEE 802.11 Wireless Local Area Network (WLAN) and Bluetooth (BT). In another embodiment, a GPS receiver is simultaneously instantiated with a cellular telephone or personal communication system (PCS) mobile terminal transceiver for GPS-enabled emergency 9-1-1 communication.

Please replace paragraph [0063], with the following rewritten paragraph:

[0063] Typically, the PCI adapter **260**, RISC CPU **265**, crossbar adapter **280**, crossbar switch **290**, reconfigurable transceivers **295**, **300**, adaptive receiver **305**, and the transceivers **310-325** are embedded within a programmable logic device such as a field programmable gate array (FPGA). In one ~~a preferred~~ implementation, the transceivers **310-325** employ a serial low

voltage differential signaling (LVDS) physical (PHY) layer. These transceivers may be implemented on a Xilinx Virtex II Pro device such as part number XC2V7 or a comparable device. The crossbar switches **290** of the described embodiment are high speed serial switches and allow for interfacing between other processor, the crossbar switch itself, as well as various I/O internal communications. In a typical implementation, the cores for the Switch Fabric Adapter **280** and the Crossbar Switch **290** are configured in the FPGA using VHDL or Verilog firmware programming. While the FPGA is one ~~a preferred~~ implementation, it is within the scope of the present invention to incorporate any reconfigurable device. Likewise, the use of serial crossbar switches is for illustrative purpose and is not to be deemed as a limiting factor.

Please replace paragraph [0067], with the following rewritten paragraph:

[0067] In one ~~preferred~~ embodiment, the module CPU **265** may be a Java connected device configuration (CDC) or connection limited device configuration (CLDC) "hardware stack" which executes Java byte code as native mode instructions, thereby providing accelerated Java operation and reduced run time memory requirements for the RTK, SCA core framework (CF), waveform software and other software which are stored as Java byte code.

Please replace paragraph [0069], with the following rewritten paragraph:

[0069] The device configuration and download of waveforms/applications is described in further detail herein. Once the SCA RTK has been "booted", the module **200** is ready to accept firmware programming and software downloads to configure various components of the module **200** including the crossbar switch **290**, software defined radio (SDR) transceiver functions **295**, **300** and adaptive processing functions **305**. These downloads may be in the form of Verilog or VHSIC Hardware Definition Language (VHDL) or other cores for FPGA, or previously compiled executable code for microprocessors and digital signal processors. In one ~~preferred~~ embodiment, a CDC or CLDC processor core optimized for the Java language may also accept software downloads of Java class files (byte code).

Please replace paragraph [0084], with the following rewritten paragraph:

[0084] The BIOS **705** contains multiple run-time software modules. There is a Hardware Abstraction Layer (HAL) manager **710**, which provides a method for providing run-time hardware abstraction for general purpose processors (GPP), application specific integrated circuits (ASIC), field programmable gate arrays (FPGA), digital signal processors (DSP), and other hardware for SDR such as those under development by the SDR Forum and other industry organizations. With HAL, SDR applications are computing platform neutral. This function interacts through the BIOS **705** with the module's resource manager **710** to the HAL software on the remote host CPU (not shown) or network server via the network interface (not shown). In one ~~a preferred~~ embodiment, the module-specific hardware resource information is based on an open standard for HAL developed by industry organizations such as the SDR Forum or OMG. In an alternative embodiment, the HAL standard is a de-facto standard for wireless devices established by manufacturers such as Sun Microsystems for Java, Palm Computing for Palm/OS or Microsoft for Windows CE and .NET.

Please replace paragraph [0087], with the following rewritten paragraph:

[0087] An additional software module is the Security services **735**. This function manages hardware and software security kernels within the SDR module and provides minimum security services to the BIOS **705**. In one ~~the preferred~~ embodiment, security manager **735** maintains a security kernel version number and validates security checksum requests from the POST **740**, Application Download **745** and Application Factory **750** functions for verification and validation (V&V) process to confirm integrity of SDR downloads and applications. This function inspects and confirms the integrity of the SDR module's current programming.

Please replace paragraph [0090], with the following rewritten paragraph:

[0090] The Application factory **750** is a piece of the SCA Core Framework function implemented on the SDR module. In one ~~a preferred~~ embodiment, SCA applications, such as waveform class files in Java, are pre-compiled by a master application factory running on the host CPU and launched by the application factory **750** on the SDR module. The application

factory launches the application using resources, file services and security services through the BIOS **705**. In another ~~a preferred~~ embodiment, the application factory **750** employs middleware such as a lightweight object request broker (ORB) or Java remote method invocation (RMI). In an alternative embodiment, the application factory **750** is replaced by a container application, such as a micro-browser, used to host and launch remotely compiled methods such as Java "applets".

Please replace paragraph [0093], with the following rewritten paragraph:

[0093] Software programs according to one embodiment which must be permanently stored on the module, such as "boot code" needed for module initialization from power-up, are stored in FLASH memory **270**. The boot code includes a minimum Software Communications Architecture (SCA) run time kernel, which may include a multiplicity of software components such as a POSIX compliant operating system micro-kernel, basic input/output system (BIOS), object request broker (ORB), hardware abstraction layer (HAL) application programming interfaces (API), SDR download API, radio control API, antenna control API, FPGA programming API, power-on self test (POST), and diagnostic/prognostic functions such as IEEE 1149.1 boundary scan support and thermal sensor support.

Please replace paragraph [0098], with the following rewritten paragraph:

[0098] When power is applied to the module, the module initiates a power-on self test (POST) and initiates a self-booting sequence of a Software Communications Architecture (SCA) run-time kernel (RTK). In one ~~the preferred~~ embodiment, the SCA RTK is implemented as a core of an embedded processor on a reconfigurable logic device such as a field programmable gate array (FPGA) or programmable logic device (PLD).

Please replace paragraph [0099], with the following rewritten paragraph:

[0099] The embedded processor's instruction set, registers, and data flow paths, and input/output ports support the needs of the SCA RTK. In one ~~the preferred~~ embodiment, implementation is reduced to the minimum number of gates and registers required to support the

needs of a "minimum" SCA RTK, minimizing the cost, size, and power required for its implementation.

Please replace paragraph [0100], with the following rewritten paragraph:

[0100] The module in one embodiment permanently stores the basic input/output system software (BIOS) needed by the SCA RTK, using an application specific integrated circuit (ASIC), FPGA, PLD, battery operated random access memory (RAM), FLASH RAM, programmable read-only memory (PROM), or some variant of one of these devices, and in one ~~a preferred~~ embodiment, the BIOS characteristics.

Please replace paragraph [0104], with the following rewritten paragraph:

[0104] A flowchart for one embodiment is shown in **Figure 9** illustrating a top level processing perspective of the self-booting process, the loading of a default waveform, and the download, verification, storage and loading of a new waveform. The start **900** commences with some event, such as power-up, that triggers the module CPU to execute the run-time kernel and commence execution of the CPU power-on self test (POST) **905**. There is initial check of certain criteria **910** which would indicate some internal errors or other problems that would prevent the device from functioning. The built-in tests (BIT) perform ~~performs~~ verification of those components that are testable and accessible. If the verifications step fails, an error report of the failure is reported **995** and processing ends **1005**. If the verification step passes, processing continues on to run boundary scan POST **915**. Once again, there is a verification check **920** and failure results in logging the error **995** and end of processing **1005**. If the boundary scan passes, the next step is the checksum **925** and the security checksum is validated **930**. A failure is processed as already described. If the checksum test is successful, the processing checks to identify a PnP interface **935**. The interface detection is verified **940** and failure commences as described. If there is a PnP interface detected, the system tries to establish a host interface **945** and perform certain functions such as loading SCA BIOS **960** and loading the default waveforms **955**, and passing the hardware abstraction layer (HAL) values identifying the hardware resource characteristics and availability to the host **950**. The passing of the HAL values **950** checks

whether a download is available **965.** If there is no download available the system loops until a download is available. If a download is available, a new waveform is downloaded **970** and verified **975**. The verification test **980** may result in a failure, which proceeds to the failure route of logging and reporting the error **995** and ending **1005**. If the verification test **980** is positive, the new application is stored **985** and the new waveform is loaded. Finally, the events and data are logged **1000** and the channel is ready for processing.

Please replace paragraph [0107], with the following rewritten paragraph:

[0107] There are numerous examples and variations associated with the present invention. With respect to FPGA's, the Xilinx Virtex II Pro, part number XC2VP7, may be employed to implement the self-booting SDR module, which are embedded within a field programmable gate array (FPGA). The XC2VP7 device contains eight transceivers, used to provide gigabit I/O between the digital transceivers and modems, and between the SDIC and its host device. It also contains a PowerPC 405 32-bit RISC CPU that can be programmed for Smart Radio functionality. The XC2VP7 device comes in a package as small as 23x23 mm when packaged in the FG456 flat-pack form factor. This small package fits well in ~~one preferred~~ PCMCIA packages, and contains 248 user available I/O pins.

Please replace paragraph [0108], with the following rewritten paragraph:

[0108] The crossbar switch core for the FPGA can be from many sources, including the Xilinx Crossbar Switch core, which is a programmable parameterizable custom design implementing digital cross-point switching functions on Xilinx Virtex-II™ and Virtex-II Pro™ FPGA. As described herein, one of the ~~preferred~~ implementations of the present invention employs the crossbar switch for use in digital cross-connects between processors and I/O nodes.

Please replace paragraph [0110], with the following rewritten paragraph:

[0110] <u>One</u> ~~The SCA in one~~ implementation uses <u>a</u> ~~the~~ self-booting RTK <u>that</u> provides interoperability with the SCA. An open source SCA Reference Implementation (SCARI) can be used as is known to those in the art. One implementation employs Java 2 Standard Edition

(J2SE) and CORBA 2.2. In a further implementation, the RTK includes a CDC or CLDC hardware Java 2 stack, on which SCA methods may be invoked.

Please replace paragraph [0111], with the following rewritten paragraph:

[0111] One or more waveforms may be determined by a transfer of software from a host (download), and stored within the module using a reconfigurable logic or memory device, such as FLASH random access memory (RAM). In one ~~the preferred~~ embodiment, the method of downloading and storing waveforms complies with a download application programming interface (API) standard such as the Software Communication Architecture (SCA).

Please replace paragraph [0112], with the following rewritten paragraph:

[0112] In one ~~a preferred~~ embodiment, two or more waveforms may be stored in such a way that the module may be rapidly switched from one stored waveform to another (switcher). In a further ~~the preferred~~ embodiment, the switcher method to select stored waveforms complies with a switcher API standard such as the SCA. When inserted into a Plug and Play (PnP) aware device, the module supports the host's PnP configuration.

Please replace paragraph [0113], with the following rewritten paragraph:

[0113] In operation, one or more waveforms may be determined by a transfer of software from a host (download), and stored within the module using a reconfigurable logic or memory device, such as FLASH random access memory (RAM). In one ~~the preferred~~ embodiment, the method of downloading and storing waveforms complies with a download application programming interface (API) standard such as the Software Communication Architecture (SCA). In another ~~the preferred~~ embodiment, two or more waveforms may be stored in such a way that the module may be rapidly switched from one stored waveform to another (switcher). In a further ~~the preferred~~ embodiment, the switcher method to select stored waveforms complies with a switcher API standard such as the SCA.

Please replace paragraph [0114], with the following rewritten paragraph:

[0114] Another implementation is to provide ease of operation, and when inserted into a Plug and Play (PnP) aware device, the module supports the host's PnP configuration. When power is applied to the module, the module initiates a power-on self test (POST) and initiates a self-booting sequence of a Software Communications Architecture (SCA) run-time kernel (RTK). In one ~~the preferred~~ embodiment, the SCA RTK is implemented as a core of an embedded processor on a reconfigurable logic device such as a field programmable gate array (FPGA) or programmable logic device (PLD). The embedded processor's instruction set, registers, and data flow paths, and input/output ports support the needs of the SCA RTK. In another ~~the preferred~~ embodiment, implementation is reduced to the minimum number of gates and registers required to support the needs of a "minimum" SCA RTK, minimizing the cost, size, and power required for its implementation.

Please replace paragraph [0115], with the following rewritten paragraph:

[0115] The module permanently stores the basic input/output system software (BIOS) needed by the SCA RTK, using an application specific integrated circuit (ASIC), FPGA, PLD, battery operated random access memory (RAM), FLASH RAM, programmable read-only memory (PROM), or some variant of one of these devices, and in one ~~the preferred~~ embodiment, the BIOS characteristics.

Please replace paragraph [0116], with the following rewritten paragraph:

[0116] The user of the host device may select one or more stored waveforms for instantiation in the SDR module by actuation of a single button or plurality of buttons. The button or plurality of buttons may be either actual (physical) buttons, as on a radio handset or user interface control panel, or virtual buttons on a host graphical user interface (GUI). In one ~~the preferred~~ embodiment, the radio button actuation interacts with the SDR module using an industry standard API such as the Defense Advanced Research Projects Administration (DARPA) ~~Globile~~ Global Mobile (GLOMO) Radio API.